Supply Chain Analytics

Vineeth Menon, MSc Big Data Analytics, L00163249

***Abstract*— The application of Big Data Analytics has exploded in many sectors as the amount of data available across digital platforms has increased. Like all other sectors, there have been tremendous amount of transformation in Supply Chain Management over the period due to digitization in ordering process, automated tracking of material movement, inventory etc. This has led the organization to leverage Big Data Platforms, Internet of Things and Cloud technologies to improve the efficiency in overall supply chain management.This Technical Report covers application of Big Data Analytics and Machine Learning (ML) using Databricks as a processing platform to analyze a Supply Chain dataset. The raw data set is of an E-Commerce company that deals with sporting, clothing and electronics supplies. The report explains the complete schema of datasets and explain the approach used to analyze the prime attributes. The report also explains the complete ML pipeline from selection of features essential to develop a model to determining its accuracy before deployment. All the inferences regarding the key attributes are covered in the result section that describes the accuracy of each implemented model and steps involved to improve it.**

***Index Terms*— Apache Spark, Big Data Analytics, Light Gradient Boosting Machine, Random Forest Classifier, Decision Tree Classifier, Machine Learning**

I. Introduction

With immense scope of Big Data applications in the domain of Supply Chain, this technical project explores a dataset used by DataCo Global that consists information on order, shipping and profit details for an E-Commerce company [1]. The primary motive of this project is to study the dataset and understand the rationale behind the application of data visualization and Machine Learning Classification algorithms to infer various hypothesis formulated through the dataset. The cloud service platform planned to use for executing the analytics will be Azure Databricks where as Pyspark will be the interface for Apache spark to implement Python APIs.

It is important to understand the schema of a dataset, total number of columns or dimensions involved, study distinct values in each dimensions, identifying key performance indicators and basic process knowledge before starting the processing and analysis. With 53 attributes, the project targets 'Delivery Status' of a product that can indicate if a product is delivered on time or late. It also focusses on 'Profit per Order' because the dataset covers order data, shipping details and its profit on per product in every region across the world. Hence it is essential to understand which region leads in maximum profit and minimum profit contribution. The project also focuses on studying the total 'Order Status' that is 'Pending' market wise and region wise. There are lots of other hypothesis that can be formulated but based on the definition of each attribute, these 3 have been given prime importance based on its impact in business performance improvement. The focussed goals of this project is to

1. Predict if a delivery is late or on time based on relevant features available

2. Identify the market with most pending orders. Based on it, identify the top countries in that market with maximum pending orders

3. Identify which order region and product has highest and lowest profit margin?

4. Identify the delivery status that tops the contribution in overall deliveries

For the execution of the project, it is important to understand the Big Data Platform in Apache Spark as the project is executed in Databricks using Pyspark. The project also emphasis on uses of multiple libraries and explain its impact. The attribute used for prediction is a categorical data, hence 2 classification algorithms are used. The selection of algorithms is justified through the accuracy measured post evaluation.

II. DATASET DESCRIPTION

The dataset for Supply Chain Analytics is acquired from https://data.mendeley.com/ Mendeley Data is a secured open source cloud-based repository for storing research datasets and reports. This particular data set selected for the project was used by DataCo Global for studying the application of Big Data title in the field of smart Supply Chain. The data has various version and this project will use latest version (version 5) published on March'19. It consists of 3 csv files. DataCoSupply-ChainDataset.csv is a structured data type that consist of 53 attributes and around 180000 entries of order details covering 50 category of products segmented under sporting, clothing and electronics supplies. This file consists of vital details such as expected days for shipping, actual days for shipping, total order quantity, profit per order, shipping mode etc. Second csv file is an unstructured data type with information of product details, URL and IP address. Third csv file consists metadata information defining each attributes in above mentioned files. The definition for key dimensions in the data are as follows:

1. Days for shipping (real): Actual shipping days of the purchased product.

2. Days for shipment (scheduled): Days of scheduled delivery of the purchased product.

3. Delivery Status: Delivery status of orders: Advance shipping, Late delivery, Shipping canceled, Shipping on time.

4. Late Delivery Risk: Categorical variable that indicates if sending is late (1), it is not late (0).

5. Market: Market to where the order is delivered.

6. Order Item Quantity: Number of products per order.

7. Order Status: If the order is completed or pending or on hold.

8. Category Name: Description of the product category

9. Order Item Discount: Discount value per product category.

10. Order Profit Per Order: Profit created per order.

11. Shipping Mode: The following shipping modes are presented: Standard Class, First Class, Second Class, Same Day.

12. Order state, Region, Country, Geographic locations at which order and shipping data are monitored.

These are precise numerical and categorical dimensions in the dataset. Rest are in form of string values. All the attributes will be studied further during data pre-processing stage. Fig. 1 displays the total columns in the dataframe.

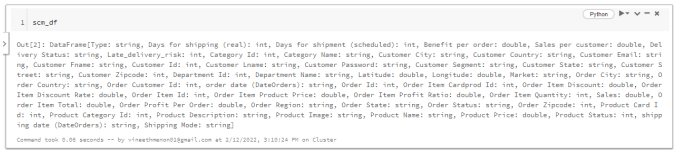


Fig. 1. Total columns in dataframe

III. BIG DATA IN APACHE SPARK

Apache spark is a popular open source engine. Spark provides fault tolerance as well as general purpose cluster computing system comprising Scala, Java, Python and R APIs for executing a task efficiently. Furthermore, Spark is well-suited for the creation of large scale machine learning systems because of its efficiency in iterative computational power.

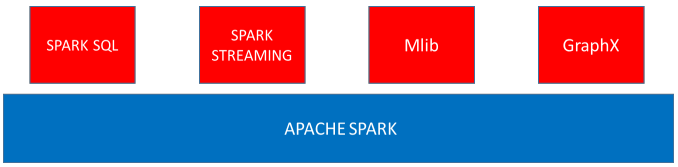


Fig. 2. Apache Spark Ecosystem

As depicted in fig2, the apache spark ecosystem consists of 4 main libraries. Spark streaming helps in real time data processing from multiple sources. GraphX is an API for graphs and graph-parallel computation. Spark SQL is a module for running SQL queries. It allows to create dataframe and act as a distributed SQL query engine. MLib helps in implementing machine learning applications in Spark. The library aims at large-scale learning environments that can benefit from data-parallelism or model-parallelism for storing and processing data and models. MLlib provides quick and scalable applications of popular learning algorithms such as classification, regression, collaborative filtering, clustering, and dimensionality reduction for common learning scenarios [3].

The main programming abstraction in Spark to execute parallel operations within a cluster are RDDs (Resilient Distributed Dataset). RDDs are created by applying various operations such as map, filter and groupBy functions to dataframes. RDD is a collection of elements partitioned across the nodes of the cluster that can be operated in parallel [4]. For this technical project, Databricks community version is used to create RDD. The cluster created in Databricks community edition consists of 2 cores, 15,5GB memory and 1 DBU (Databricks unit). The runtime version is 6.4 Extended Support, Spark version 2.4.5 and Scala version 2.11.

IV. CODE FILES

Git Hub link for all project documents can be found at `https://github.com/Vineeth08/BDA--Technical-Project-1.git`

V. DATA ACQUISITION AND PRE-PROCESSING

With the datasets being acquired and finalized, the primary step is to upload it to the Databricks File System (DBFS). This can be done through creating table and uploading the datasets directly from the path or from the desktop file location. This dataset has to be pulled in the form of a schema to read, write and create models. In Jupyter-Notebook, 'Pandas' library helps us to load the data in the form of dataframe. But with Apache Spark set up, the acquisition will be with the help of 'pyspark.sql.types' library. Go through below codes.

```
'from pyspark.sql.types import StructType,
StructField, IntegerType, StringType,
DoubleType, TimestampType

scm_df = (sqlContext.read.format("csv")'
```

StructType,StructField, IntegerType and others being imported is for Pyspark to recognize if the variable in the schema is an integer or string or float. Because of the import, the dataframe output will itself recognize the type of data in the schema. The output displays 2 jobs. The reason is that as RDD is invoked, a "job" is created. Jobs are work submitted to Spark. Jobs are divided into "stages" based on the shuffle boundary. Refer below figure.Each stage is further divided into tasks based on the number of partitions in the RDD. So tasks are the smallest units of work for Spark [7].

To view the schema, use command

```
display(scm_df)
```

following table with header will be displayed.
To check number of rows and columns,

```
'print((scm_df.count(), len(scm_df.columns)))'.
```

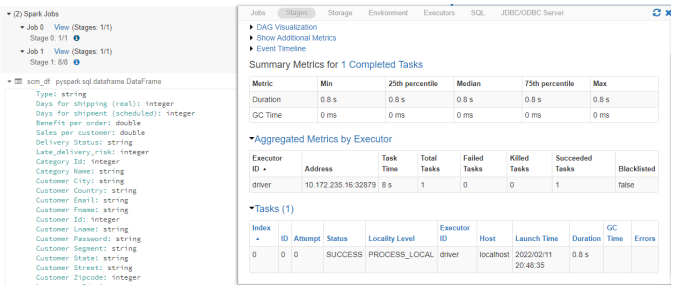


Fig. 3. Run output displaying 2 jobs, stages and tasks

The output indicates that the dataset possesses 180519 rows and 53 columns. This is to understand and study the data to recognize what further steps need to be taken. This is different from python in Jupyter Notebook where rows and columns are displayed using .shape() function.

Data pre-processing is done to have a smooth executing of ML pipeline with best accuracy. Does it should not possess any null values. So it's necessary to check if current possess any null values. 3 columns possess null values. Product description – 180519 Order Zipcode – 155679 Customer Zipcode – 3 Data in all these columns are not significant to run algorithms. Hence, drop Product description, Order Zipcode. Where as using na.fill, enter '0' for 3 values of Customer Zipcode. The new dataframe is named as scm_df2.

VI. EXPLORATORY ANALYSIS

It is important to visualize the dataframe to understand the data, its trend, pattern etc. Many questions can be answered just through mere data visualization. Good data visualization makes it easier to communicate information to larger crowd. The visualization has been given prime importance in this project. The target metrics for visualization are 'Order Status' for various markets. This will be further broken down to region wise and country wise. 'Delivery Status' region wise and 'Order Profit per Order'. All the attributes have been defined above. It is understood that with the given dataset, the most important metrics that can impact business are pending orders, late delivery and less order profit. Refer below graphs.

The visualization targets the regions and countries where improvements should be initiated to improve all 3 parameters.

The fig.4 explains 'Market with Pending Status' in every market. This illustrates total orders pending as per 5 distinct market regions. Considering the organization needs to work against reducing the pending orders, a bar chart in descending order will help in prioritising the markets and its respective countries to take actions for reducing the total pending orders.

For visualization, the project imports 3 important libraries namely matplotlib, seaborn and plotly. Seaborn library is used for data visualization such as scatter plots, checking data normality etc. Matplotlib is another library used for visualization. It embeds many other visualization library (such as seaborn). It is very handy while working in multidimensional arrays. %Matplotlib Inline provides backend support to matplotlib. It will register a function that gets triggered whenever the output of a cell is a figure.

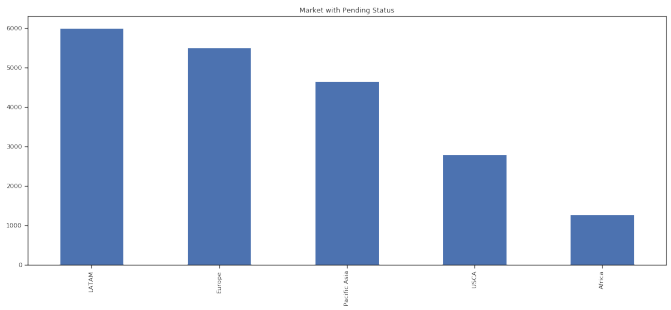


Fig. 4. Market with Pending Status

From the bar plot, its noted that Latam market has maximum pending orders. Hence, lets explore which countries within Latam has maximum pending orders.

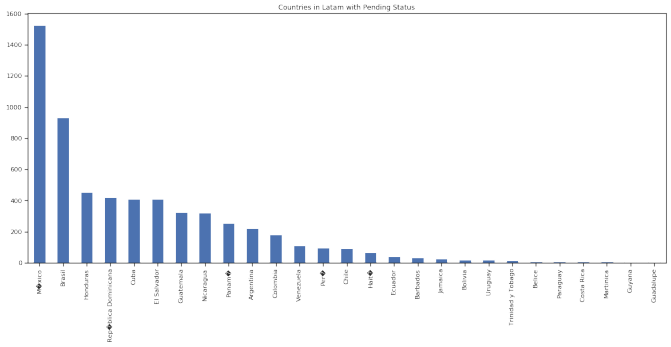


Fig. 5. Countries in Latam with Pending Status

It is learnt that Mexico and Brazil contributes to highest volume of pending orders.

For the next hypothesis on Order profit region wise and product wise, a horizontal bar plot will be used due to large number of distinct values.

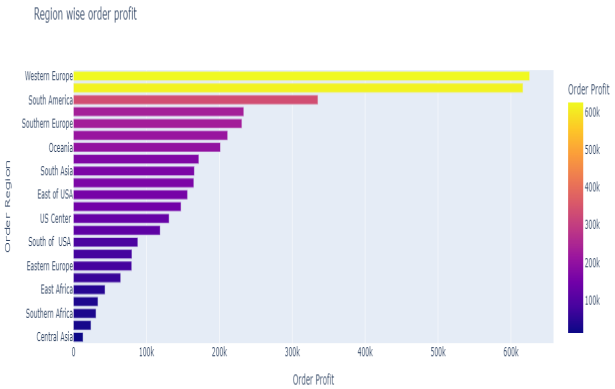


Fig. 6. Region wise order profit

Western Europe with $625446 and Central America with 616341.

Fishing category products and Cleats category products have highest profit margin, where as the organization needs to work upon increasing order profit for Soccer, Strength training and Video games going with the popularity.

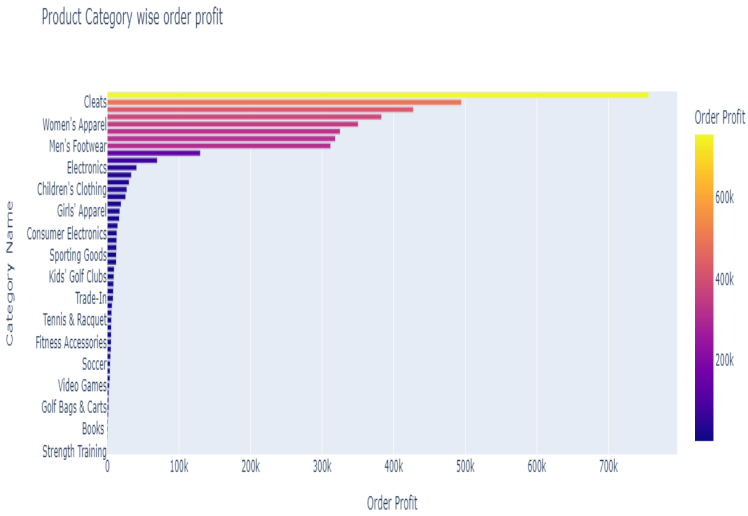


Fig. 7. Product Category wise order profit

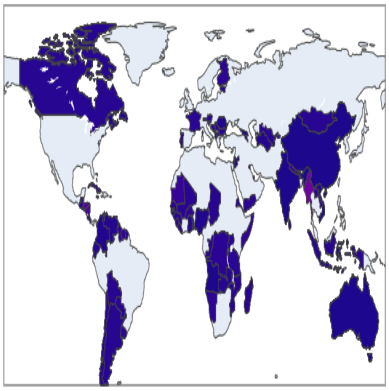


Fig. 8. Plotting order profit country wise using geographic plot

The scatter plot between Discount rate vs Sales was plotted to notice if there is any visible relation between sales to the discount rate. However, the dispersed points in scatter plot does not indicate any strong correlation.

Delivery status graph in Fig.10 is plotted to notice the impact of late delivery on overall status. It can be seen that 'Late Delivery' is a major concern for all regions and markets. Central America and Western Europe has highest late deliveries owing to the fact that number of orders are high too. Therefore, it is essential to create a model to predict the late deliveries so that it can be controlled based on vital features.

Thus through visualization, one can interpret the behaviour

Fig. 9. Scatter Plot: Order Item Discount Rate vs Sales

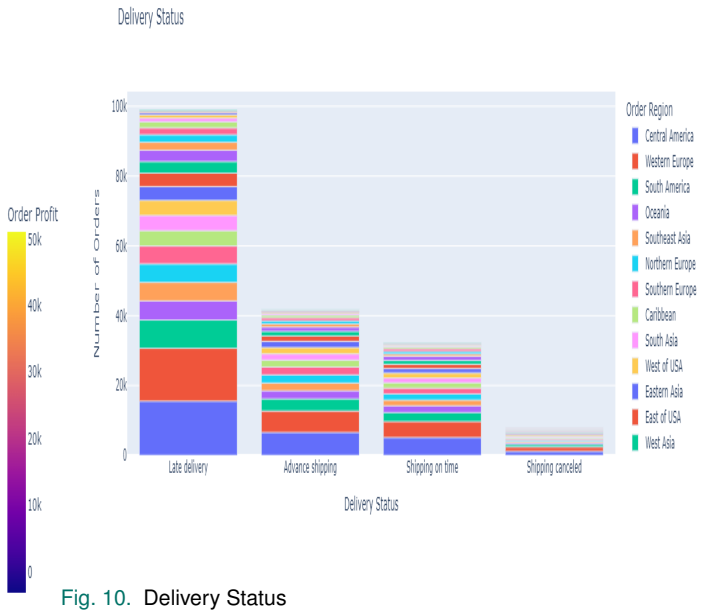


Fig. 10. Delivery Status

and patterns of key performance indicators. Post visualization and data processing, the finalized dataset can be used for developing ML models.

VII. Creating Pipeline and Model Selection

The flow diagram (Fig.11) illustrates the number of steps involved in creating a complete machine learning pipeline. The data transformed earlier cannot be directly used for creating models. Identifying features to fit in a model is essential.

A. Training and Testing Dataset

Once a dataframe has been finalized after transformation, its needed to be split for training and testing. The proportion for the split is by default 80 % and 20 %, while there are different school of thoughts on selecting the actual proportion. An ideal split will avoid overfitting and under fitting of model. Some articles would tell us to use a 70:30 or even a 50:50 split. All of it depends on how large is a dataset. In this project, a trial

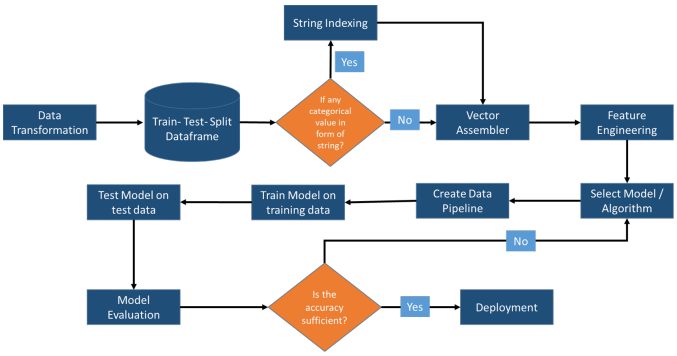


Fig. 11. Machine Learning Pipeline

```
from pyspark.ml.feature import VectorAssembler
assembler = VectorAssembler(
  inputCols = [
    "Benefit per order",
    "Sales per customer",
    "Indexed_Type",
    "Indexed_Market",
    "Indexed_Shipping Mode",
    "Indexed_Department Id",
    "Indexed_Customer Segment",
    "Indexed_Category Name",
    "Order Item Discount",
    "Order Item Discount Rate",
    "Order Item Profit Ratio",
    "Sales"],
  outputCol = "features")
```

Command took 0.04 seconds -- by vineethmenon01@gmail.com at 2/12/2022, 4:03:28 PM on Cluster

Fig. 12. Feature Engineering using Vector Assembler

and error will be taken on split ratio between 70:30 and 80:20 as per accuracy of the output.

The training data set is used to fit the model. Hence larger the data size better the model outcome. The test set is used for an fair evaluation of the final model.

B. Feature Engineering:

Feature engineering is a process where raw data is converted into features such that it can be used in machine learning algorithms. Not all data has to be transformed to features. In order to identify the exact features, domain expertise is required. The plan of model development in this dataset is to predict whether a delivery is on time or late. The label indexer used for output is the values in column `Late_Delivery_Risk`. Hence selection of features will be based on parameters that are relevant to accurate prediction of `Late_Delivery_Risk`. For feature engineering, 2 main modules needed are 'VectorAssembler' and 'StringIndexer'. Both are imported from pyspark.ml libraries.The role of a vector assembler is to transform multiple columns into one single vector column. The transformation is done as follows: 'class pyspark.ml.feature.VectorAssembler(inputCols= " ", outputCol= "features")' The name of the output column is "features". Now, some columns would have categorical data in form of strings. Machine learning algorithm will not function in presence of strings, hence it has to be converted into index or numericals using StringIndexer or OneHotEncoder. These indexed values are also added as input columns into VectorAssembler for feature engineering. Hence total 12 columns are added as inputs for feature column. Please note that additional features will be added and removed based on model accuracy. Refer GitHub link code files and below figure.

C. Selection of Algorithm (Model)

Selection of model depends on type of data that is needed to be predicted. In feature Engineering, feature for label column is selected as `Late_Delivery_Risk`. The supervised learning algorithms are divided into 2 categories. Regression and Classification algorithms. The main difference in application of these 2 categories are that classification algorithms are used when the value to be predicted are discrete values (labels) where as regression algorithms are used when the output is continuous data or some measurable quantity. The label column selected for prediction has binary output. If result is 1 then delivery is late and if result is 0 then result is on time shipping or advanced shipping. The purpose of developing this model is to identify the factors (features) that will lead to late delivery shipping so that it can be controlled for least late deliveries. Since the value to be predicted is binary, this project explores possibilities of classification algorithms. In classification division, 2 algorithms selected for the project are Light Gradient Boosting Machine and Random Forest classificaion model. One can technically use a list of algorithms and finalise the best based on accuracy rates.

D. Light Gradient Boosting Machine

Extreme Gradient Boosting (XGBoost) and Gradient boosting decision tree (GBDT) are highly recommended classifications for analyzing the label because they offer the best mix of accuracy, stability, and computational efficiency. Light Gradient Boosting Machine (LightGBM) is an upgraded version of "gradient learning framework" based on "decision trees" and the idea of "weak" learners. Its developed by Microsoft in 2017. The difference between LGBM and XG Boost model is that the latter uses histogram-based algorithms to increase the speed up the training model, reduce memory consumption and deploys a leaf wise growth strategy with depth constraints [5].

Due to the inefficiency in terms of growth strategy, the decision tree is a poor model. It only handles the leaves of the same layer, resulting in a significant amount of wasted memory. Large number of leaf wise level can cause overfitting. LightGBM provides a "maximum depth limit" to the top of the leaf to avoid overfitting that enables high efficiency. In the project, mml Spark library is installed using Maven coordinates. MML spark library possess packages for most of the classification and regression models. Import LightGBMClassifier. The featuresCol will be "features" derived from Vector Assembler and labelCol=`Late_Delivery_Risk`. Number of iterations considered is 100.

E. Random Forest Classifier

Random forest is an example of Ensemble Learning. Ensemble learning is a method where multiple models try and solve one problem. It's an ensemble of Decision trees and decides class type through voting. Due to this Random Forest tends to

Fig. 13. Running LightGBMClassifier using Pyspark

have high accuracy than many classifiers. Random Forest (RF) use bagging method with a twist. In bagging, splitting happens as per Bootstrap that can lead to selection of similar features. But in RF, splitting happens randomly instead of splitting at similar feature [6]. While implementing the model, it's necessary to mention number of trees. Does like LightGBM, RF classifier is imported from mmlSpark library. The creation of pipeline is similar to LightGBMClassifier. featuresCol and labelCol is same. After testing the accuracy, features will be added or removed.

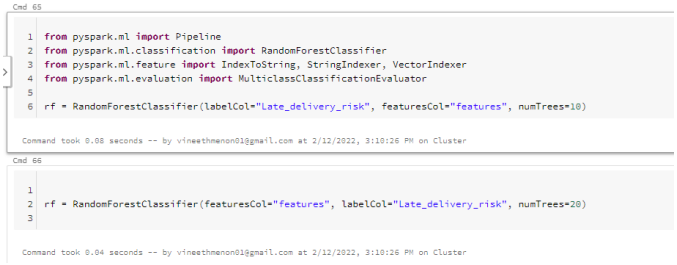


Fig. 14. Running RandomForestClassifier using Pyspark

F. Model Evaluation

This is the most critical stage in the project. For evaluating the model, 'MulticlassClassificationEvaluator' has to be imported from pyspark.ml.evaluation library. The evaluation results for each model and further actions are covered in result section. The evaluator provides accuracy rate between the actual values and predicted values. Based on accuracy needed, correct model is selected for deployment.

VIII. Results

Based on the prediction model run above, The LGBM Classifier provided accuracy up to 69%. This is a good accuracy but needed to be improved. Hence Random Forest Classifier model is used to recheck if the accuracy can be improved. Refer Fig. 15 and Fig. 16

The accuracy of Random Forest Classifier was also approximately 69%. Hence to check if accuracy can be improved, more features were added. 28 features are added in vector assembler that consists of 17 numerical features and 11 indexed features. The newly added features consisted of Order Regions, Product Categories, Product Names, Departments etc.

Post creating pipeline with LightGBMClassifier with all new features, the accuracy turned out to be 100%.

Even though the accuracy is best, it should be verified with other classifiers. The Random Forest needed a huge

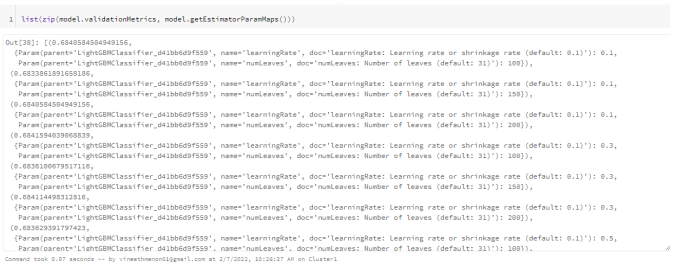


Fig. 15. Light Gradient Boosting Machine Classifier Accuracy with ParamGridBuilder()

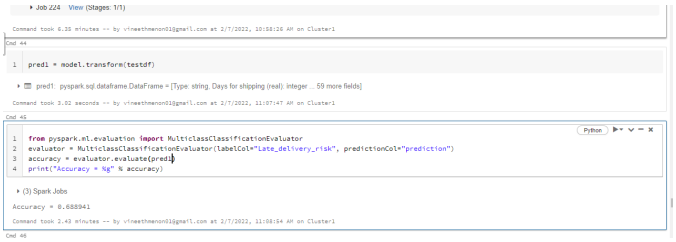


Fig. 16. Random Forest Classifier Accuracy

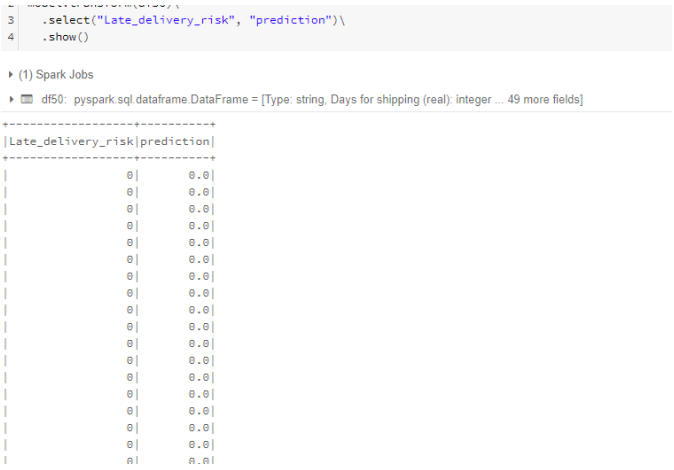
```
from pyspark.ml.feature import VectorAssembler
feature_list = ['Indexed_Type',
 'Days for shipping (real)',
 'Days for shipment (scheduled)',
 'Benefit per order',
 'Sales per customer',
 'Indexed_Category Name',
 'Indexed_Customer City',
 'Indexed_Customer Country',
 'Indexed_Customer Segment',
 'Department Id',
 'Latitude',
 'Longitude',
 'Indexed_Market',
 'Order Item Discount',
 'Order Item Discount Rate',
 'Order Item Product Price',
 'Order Item Profit Ratio',
 'Order Item Quantity',
 'Sales',
 'Order Item Total',
 'Order Profit Per Order',
 'Indexed_Order Region',
 'Indexed_Order State',
 'Indexed_Order Status',
 'Indexed_Product Name',
 'Product Price',
 'Product Status',
 'Indexed_Shipping Mode']

assembler = VectorAssembler(inputCols=feature_list, outputCol="features")
```

Fig. 17. Feature Engineering with additional attributes

Fig. 18. Light Gradient Boosting Machine Classifier Accuracy with new features

amount of extra training data for running all categorical values.



```
    .select("Late_delivery_risk", "prediction")\
    .show()
```

▸ (1) Spark Jobs

▸ df50: pyspark.sql.dataframe.DataFrame = [Type: string, Days for shipping (real): integer ... 49 more fields]

Late_delivery_risk	prediction
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0
0	0.0

Fig. 19. Actual vs Prediction

The Actual vs Predication table indicates the comparison between actual values from 20% testing dataset and predicted values as per LGBM Classifier where where 1 is Late Delivery and 0 is Shipping on Time.

IX. Conclusion

Thus to summarise the hypothesis:

1. Predict if a delivery is late or on time based on relevant features available - LightGBMClassifier can be used as a prediction model with adequate features. The accuracy first came out to 69% and then 100%. More data can be collected to test if the accuracy is persistent.

2. Identify the market with most pending orders. Based on it, identify the top countries in that market with maximum pending orders - It is understood that Mexico and Brazil from Latam market possess maximum pending orders. These countries need to work on scheduling to reduce the overall count and customer satisfaction.

3. Identify which order region and product has highest and lowest profit margin - Western Europe with $625446 and Central America with $616341

Fishing category products - $756220

Cleats category products - $494636

Need to work upon increasing order profit for Soccer, Strength training and Video games going with the popularity

4. Identify the delivery status that tops the contribution in overall deliveries - Central America and Western Europe has maximum late deliveries (over 15000 orders). Substantial improvement is required.

The dataset can undergo many other hypothesis and models. The aim from the following project is to understand how to approach a data in real world scenario and draw inferences that can improve the efficiency of a process. For an E-commerce company, many similar daily transactional data can be extracted and used for Online Analytical Processing using various visualization tools.

References

[1] Constante, F, F Silva, and A Pereira. “DataCo smart supply chain for big data analysis” Mendeley Data, v5 (2019): 1-13.

[2] Alicke, Knut, Daniel Rexhausen, and Andreas Seyfert. ”Supply Chain 4.0 in consumer goods.” Mckinsey Company 1, no. 11 (2017).

[3] Meng, Xiangrui, Joseph Bradley, Burak Yavuz, Evan Sparks, Shivaram Venkataraman, Davies Liu, Jeremy Freeman et al. ”Mllib: Machine learning in apache spark.” The Journal of Machine Learning Research 17, no. 1 (2016): 1235-1241.

[4] Ilijason, Robert. Beginning Apache Spark Using Azure Databricks: Unleashing Large Cluster Analytics in the Cloud. Apress, 2020.

[5] Fan, Junliang, Xin Ma, Lifeng Wu, Fucang Zhang, Xiang Yu, and Wenzhi Zeng. ”Light Gradient Boosting Machine: An efficient soft computing model for estimating daily reference evapotranspiration with local and external meteorological data.” Agricultural Water Management 225 (2019): 105758.

[6] Dogru, Nejdet, and Abdulhamit Subasi. ”Traffic accident detection using random forest classifier.” In 2018 15th learning and technology conference (LT), pp. 40-45. IEEE, 2018.

[7] n.d. ”RDD Programming Guide.” Spark 3.2.1